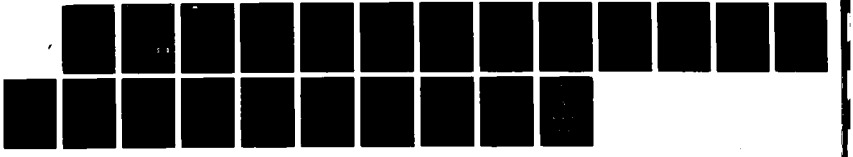
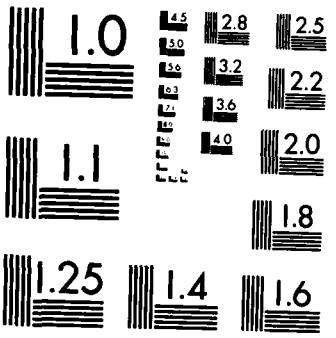
AD-A188 307 ALGEBRAIC METHODS APPLIED TO NETWORK RELIABILITY PROBLEMS REVISION(U) CLEMSON UNIV SC DEPT OF MATHEMATICAL SCIENCES D R SHIER ET AL. SEP 86 1/1

UNCLASSIFIED TR-486-REV AFOSR-TR-87-1589 AFOSR-84-0154 F/G 12/4 NL





(2)

AFOSR-TR- 87-1589

ALGEBRAIC METHODS APPLIED TO NETWORK RELIABILITY PROBLEMS

D. R. Shier

D. E. Whited

Department of Mathematical Sciences

Clemson University

Technical Report #486

August, 1985

September, 1986 (Revised)

This research was supported by the U.S. Air Force Office of Scientific Research (AFSC) under Grant AFOSR-84-0154

UNCLASSIFIED
SECURITY CLASSIFICATION [redacted]E

AD-A188 307

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY NA	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for Public Release; Distribution Unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE NA	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) AFOSR-TR. 87-1589
6a. NAME OF PERFORMING ORGANIZATION Clemson University 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION AFOSR/NM
6c. ADDRESS (City, State, and ZIP Code) Clemson SC 29634-1907	7b. ADDRESS (City, State, and ZIP Code) Bldg. 410 Bolling AFB, DC 20332-6448
8a. NAME OF FUNDING / SPONSORING ORGANIZATION AFOSR 8b. OFFICE SYMBOL (If applicable) NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR-84-0154
8c. ADDRESS (City, State, and ZIP Code) Bldg. 410 Bolling AFB, DC 20332-6448	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 6-1102F; PROJECT NO. 2304; TASK NO. A5; WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Algebraic Methods Applied to Network Reliability Problems (UNCLASSIFIED)

12. PERSONAL AUTHOR(S)
D. R. Shier and D. E. Whited

13a. TYPE OF REPORT ~~Technical Report~~ Reprint	13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year, Month, Day) September, 1986	15. PAGE COUNT 18

16. SUPPLEMENTARY NOTATION
Department of Mathematical Sciences, Technical Report #486

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Algorithm; approximation; iterative; reliability

ABSTRACT (Continue on reverse if necessary and identify by block number)

An algebraic structure underlying network reliability problems is presented for determining the 2-terminal reliability of directed networks. An iterative algorithm is derived from this algebraic perspective to solve the (s,j)-terminal reliability problem simultaneously for all nodes j. In addition to providing an exact answer (in the form of a reliability polynomial), the algorithm also yields a nondecreasing sequence of approximate solutions guaranteed to be lower bounds on the exact solution. Empirical results, presented for two different implementations of the algorithm, show that useful approximate solutions can be obtained in a reasonable amount of computation time.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Brian W. Woodruff, Maj.	22b. TELEPHONE (Include Area Code) (202) 767-5027 22c. OFFICE SYMBOL AFOSR/NM

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

ALGEBRAIC METHODS APPLIED TO NETWORK RELIABILITY PROBLEMS

Douglas R. Shier

David E. Whited

Abstract. An algebraic structure underlying network reliability problems is presented for determining the 2-terminal reliability of directed networks. An iterative algorithm is derived from this algebraic perspective to solve the (s, j)-terminal reliability problem simultaneously for all nodes j. In addition to providing an exact answer (in the form of a reliability polynomial), the algorithm also yields a nondecreasing sequence of approximate solutions guaranteed to be lower bounds on the exact solution. Empirical results, presented for two different implementations of the algorithm, show that useful approximate solutions can be obtained in a reasonable amount of computation time.

Key words. algorithm, directed graphs, reliability

AMS (MOS) subject classifications: 05C20, 62N05, 94C15

1. Introduction

The problem of determining the reliability of an existing or proposed communication system has received considerable attention in the engineering, statistical, and operations research literature [1, 5, 9]. For example, it is important to assess the probability that a message sent from a given source arrives at its destination, when the components comprising the system are subject to failure. Unfortunately, most reliability problems of any substance are now known to be NP-hard or #P-complete [4, 17, 18, 27]. As a result, researchers have focused on special network structures (where polynomial-time algorithms are possible), or have resorted to simulation.

A number of special classes of undirected networks have recently been analyzed with success. Polynomial-time algorithms are now available for calculating certain reliability measures in series-parallel [21], inner-cycle-free [14], inner-four-cycle-free [14], and cube-free [15] planar graphs. Provan [16] has shown, however, that the problem of determining source-to-terminal reliability remains #P-complete for the general class of planar graphs. In order to analyze more complex network topologies, the idea of pivotal decomposition [5] together with polygon-to-chain reductions [28] can be used to decompose the original problem into smaller subproblems.

Similar results and tools are not as available in the case of directed networks. The only significant types of directed networks that are known to admit a polynomial-time algorithm are "basically-series-parallel" networks [2, 3]. Also, unlike the case for undirected networks, certain simplifications are not available when carrying out pivotal decomposition in the directed case [1]. Nor does there exist an "optimal" factoring algorithm, such as that demonstrated for undirected networks [19].

This paper exploits the underlying algebraic structure of network reliability problems to produce a general iterative algorithm, applicable to arbitrary directed networks. While not polynomially-bounded, it is able to generate reasonable

approximations to exact network reliability with a modest amount of computation.

2. Algebraic Structure

Suppose that G = (N, E) is a directed network with node set N and edge set E, in which nodes do not fail but edges fail independently of one another. The *reliability* of edge e (the probability that edge e functions) is denoted by p_e. Nodes s and t designate the specified source and terminal of G, and we are interested in calculating the 2-terminal reliability $R_{st}(G)$:

$$R_{st}(G) = \Pr\{\text{there exists a functioning path from s to t in G}\}.$$

Associate with each edge $k \in E$ a *variable* x_k. Then the *reliability polynomial* $F_{st}(\mathbf{x}) = F_{st}(x_1, ..., x_r)$ associated with s and t is a polynomial in $x_1, ..., x_r$ such that if the numerical values $p_1, ..., p_r$ are substituted for the corresponding variables $x_1, ..., x_r$ then the resulting value is the probability that a functioning path exists from node s to node t. (If the x_i's were simply Boolean variables, this polynomial would be identical with the *structure function* of the system [5].) The reliability polynomial can be concisely expressed using two operations $\oplus$ and $\otimes$ defined on polynomials.

To begin, let

$$T^a = x_1^{a_1} x_2^{a_2} \cdots x_r^{a_r}$$

denote a monomial *term*, where each $a_i \in \{0,1\}$. The operation $\otimes$ when applied to terms T^a and T^b yields the term T^c where $c_i = \max\{a_i, b_i\}$. This operation is extended to arbitrary polynomials by distributivity. The operation $\oplus$ is defined on polynomials $f(\mathbf{x})$ and $g(\mathbf{x})$ using

$$f(\mathbf{x}) \oplus g(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) - f(\mathbf{x}) \otimes g(\mathbf{x}).$$

Operations related to $\oplus$ and $\otimes$ were apparently first suggested by Mine [13] and by

Kim et al. [10]. More recently Gondran and Minoux [8], and Shier [24], have formulated network reliability using the operations $\oplus$ and $\otimes$ defined above.

Let S denote the set of all polynomials that can be formed from monomial terms T^a, T^b, ... by finite applications of the operations $\oplus$ and $\otimes$. Then it can be demonstrated [24] that $(S, \oplus, \otimes)$ forms a distributive lattice with smallest element 0 (the zero polynomial) and largest element 1 (the unit polynomial). Suppose that P_{st} is the set of simple paths from node s to node t in G. Define the *value* v(P) of path P to be the product, with respect to $\otimes$, of the edge variables along the path:

$$v(P) = \otimes \prod \{x_k : k \in P\}.$$

Then the reliability polynomial $F_{st}(\mathbf{x})$ can be expressed as

$$F_{st}(\mathbf{x}) = \oplus \sum \{v(P): P \in P_{st}\} . \tag{1}$$

As an illustration, consider the standard bridge network in Fig. 1 having s = 1 and t = 4. Since there are four simple paths extending from s to t, equation (1) becomes

$$F_{st}(\mathbf{x}) = x_1x_4 \oplus x_1x_3x_5 \oplus x_2x_4x_6 \oplus x_2x_5 .$$

Equation (1) is just the standard expression for the inclusion-exclusion formula, applied to paths in the network [1]. Expanding such an expression, using the definitions of $\oplus$ and $\otimes$, and then substituting numerical values p_k for the corresponding variables x_k yields $R_{st}(G)$.

A number of techniques have been developed to calculate the quantity (1) for a general algebraic structure $(S, \oplus, \otimes)$ satisfying appropriate properties [6, 8]. These techniques can all be viewed as different methods of solving the system of equations

$$\mathbf{z} = \mathbf{z}\, M \oplus \mathbf{e}_s, \tag{2}$$

which is linear in the operations $\oplus$ and $\otimes$. Here $M = (m_{ij})$ is the *weighted adjacency matrix* for G, with $m_{ij} = x_k$ for $k = (i, j) \in E$ and $m_{ij} = 0$ otherwise. Also, $\mathbf{e}_s$ denotes the s-th unit row vector. An (extremal) solution $\mathbf{z}$ to these equations is known [6] to satisfy

$z_j = F_{sj}(\mathbf{x})$ for all $j \in N$. Thus by solving such equations to find $F_{st}(\mathbf{x})$, and hence $R_{st}(G)$, we also obtain the (s, j)-terminal reliabilities for all $j \in N$. Moreover, unlike existing methods for calculating 2-terminal network reliability based on paths [1, 11], such algebraic methods do not need to first enumerate all simple paths joining the two terminals [6, 8]. These paths are automatically generated in the course of solving the set of equations (2).

A natural way of solving (2) is by means of an iterative procedure, whereby the current estimate for **z** is substituted into the right-hand-side of (2), producing a new estimate for the solution vector **z**. In the next section, we discuss a specific iterative scheme for solving (2) that incorporates special data structures to streamline such computations.

3. An Iterative Scheme

The basic idea of the iterative scheme presented here is that of passing on, at each step, the information available at node i to each of its *neighbors* j, where $(i, j) \in E$. Before stating the general iterative scheme, the ideas will first be illustrated using the network in Fig. 1. We will find all reliabilities $z_j = F_{sj}(\mathbf{x})$ relative to the source node s = 1.

In the algorithm, a polynomial *label* is associated with each node j. At any stage, LABEL(j) will be a reliability polynomial based on a certain subset of paths from node s to node j. In this sense, LABEL(j) corresponds to a current estimate of the solution z_j to equation (2). Initially, if there is an edge $k = (s, j) \in E$ then LABEL(j) = x_k. If there is no such edge then LABEL(j) = 0; in the case of the source node, LABEL(s) = 1. Those nodes, apart from s, receiving a nonzero initial label are placed on a list L. In this example, we have

j:	1	2	3	4
LABEL(j):	1	x_1	x_2	0
L:	[3, 2]			

Now we remove the "top" node i from L and update its neighbors j using

(3) $$\text{LABEL}(j) := \text{LABEL}(j) \oplus [\text{LABEL}(i) \otimes x_k] ,$$

where k = (i, j). The above *(i, j) update* simply incorporates into LABEL(j) new paths from s to j that use the edge (i, j). Any node j whose label is changed by (3) is placed on L if it does not already appear. This step removes, in our example, i = 3 and updates

$$\text{LABEL}(2) = x_1 + x_2x_6 - x_1x_2x_6$$

$$\text{LABEL}(4) = x_2x_5$$

$$L = [2, 4]$$

The corresponding network, with node labels attached, is shown in Fig. 2.

At the next step, node i = 2 is removed from the top of L. Nodes 3 and 4 are then updated, and node 3 is added to L:

$$\text{LABEL}(4) = x_2x_5 + x_1x_4 - x_1x_2x_4x_5 + x_2x_4x_6 - x_2x_4x_5x_6 - x_1x_2x_4x_6 + x_1x_2x_4x_5x_6$$

$$\text{LABEL}(3) = x_2 + x_1x_3 - x_1x_2x_3 + x_2x_3x_6 - x_2x_3x_6 - x_1x_2x_3x_6 + x_1x_2x_3x_6$$

$$= x_2 + x_1x_3 - x_1x_2x_3$$

$$L = [4, 3]$$

This process is continued until L becomes empty. At this point, the polynomial label on any node j represents $z_j = F_{sj}(\mathbf{x})$. Table 1 shows the final labels for our example, together with the value obtained by substituting the common edge reliability p for all x_k.

Table 1

	$F_{sj}(x)$	$R_{sj}(G),\ p_k = p$
j = 1	1	1
j = 2	$x_1 + x_2x_6 - x_1x_2x_6$	$p + p^2 - p^3$
j = 3	$x_2 + x_1x_3 - x_1x_2x_3$	$p + p^2 - p^3$
j = 4	$x_2x_5 - x_1x_2x_4x_6 + x_1x_2x_4x_5x_6$ $+ x_2x_4x_6 - x_2x_4x_5x_6 + x_1x_4$ $- x_1x_2x_4x_5 + x_1x_3x_5 - x_1x_3x_4x_5$ $- x_1x_2x_3x_5 + x_1x_2x_3x_4x_5$	$2p^2 + 2p^3 - 5p^4 + 2p^5$

The general form of the iterative procedure is specified by the following algorithm, where L again represents the list of nodes whose labels have been changed.

1. [Initialization]

   for $j \neq s$ **do**

   If $k = (s, j) \in E$ **then** LABEL(j) := x_k

   else LABEL(j) := 0;

   LABEL(s) := 1;

   L := [j: $(s, j) \in E$];

2. [Iterative Step]

while $L \neq [\ ]$ **do**

remove i from L;

for $k = (i, j) \in E$ **do**

$T := \text{LABEL}(j) \oplus [\text{LABEL}(i) \otimes x_k]$;

if $T \neq \text{LABEL}(j)$ **then**

$\text{LABEL}(j) := T$;

if $j \in L$ **then** enter j into L.

Upon termination of the algorithm, LABEL(j) will be the required reliability polynomial $F_{sj}(\mathbf{x})$. Notice that there are several ways of managing the list L. In our example, we treated L as a queue, whereby nodes are processed in a FIFO (first-in-first-out) manner. It is also possible to treat L as a stack, whereby nodes are processed in a LIFO (last-in-first-out) manner. The effect of these two ways of managing L will be examined in Section 5. First, we discuss a number of useful properties of this iterative algorithm.

4. Properties

In this section we make use of the algebraic properties of $(S, \oplus, \otimes)$ to establish certain properties of the iterative algorithm presented in Section 3. It will be convenient to denote the variable attached to edge (i, j) by x. Also, the label on node j at the start of step m will be denoted by $L_m(j)$. Then the (i, j) update (3) of node j after step m is expressed as

$$L_{m+1}(j) = L_m(j) \oplus x\, L_m(i). \tag{4}$$

Because the label on node j represents the sum with respect to $\oplus$ of a set of simple

s-j paths and because this set of paths can expand through subsequent updates (4), we have

Property 1. If $k \leq m$ then $L_k(j) \oplus L_m(j) = L_m(j)$.

One important simplification derives from the following property. It states that only the "new" information N(i) added to the label of i since i was last on L needs to be propagated to its neighbors j.

Property 2. Suppose that at step m an (i, j) update is to be performed, where the labels on i and j are $L_m(i) = L_k(i) \oplus N(i)$ and $L_m(j)$ with $k < m$. Step k represents the step at which an (i, j) update previously occurred. Then at step m+1 the new label assigned to j will be $L_{m+1}(j) = L_m(j) \oplus x\ N(i)$.

Proof. At step k, node j receives the label $L_{k+1}(j) = L_k(j) \oplus x\ L_k(i)$. Also, since $k < m$ we have $L_{k+1}(j) \oplus L_m(j) = L_m(j)$, by Property 1. Then

$$
\begin{aligned}
L_{m+1}(j) &= L_m(j) \oplus x\ L_m(i) \\
&= [L_m(j) \oplus L_{k+1}(j)] \oplus x\ [L_k(i) \oplus N(i)] \\
&= [L_m(j) \oplus L_k(j) \oplus x\ L_k(i)] \oplus x\ L_k(i) \oplus x\ N(i) \\
&= [L_m(j) \oplus L_k(j) \oplus x\ L_k(i)] \oplus x\ N(i) \\
&= [L_m(j) \oplus L_{k+1}(j)] \oplus x\ N(i) \\
&= L_m(j) \oplus x\ N(i)\ .\ \blacklozenge
\end{aligned}
$$

Because the labels on each node will be maintained as fully expanded polynomials (expressed using ordinary + and ×), it is desirable to know when certain of the terms in $L_m(i)$ do not affect the label $L_m(j)$. The following property provides one such condition.

Property 3. Suppose that at step m an (i, j) update is to be performed with $L_m(i) = A_1 + A_2 + \ldots + A_v$, $L_m(j) = B_1 \oplus B_2 \oplus \ldots \oplus B_w$ and $B_1 \subseteq xA_1$. Then the updated label $L_{m+1}(j) = L_m(j) \oplus x\ [A_2 + \ldots + A_v]$.

Proof. Let $A = A_2 + \ldots + A_v$ and $B = B_2 \oplus \ldots \oplus B_w$. Then

$$L_{m+1}(j) = L_m(j) \oplus x\, L_m(i)$$
$$= [B_1 \oplus B] \oplus x\,[A_1 + A]$$
$$= [B_1 \oplus B] + [xA_1 + xA] - [xA_1 + xA]\,[B_1 \oplus B]$$
$$= [B_1 \oplus B] + xA_1 + xA - xA_1[B_1 + B - B_1B] - xA\,[B_1 \oplus B]$$
$$= [B_1 \oplus B] + xA_1 + xA - xA_1 - xA_1B + xA_1B - xA\,[B_1 \oplus B]$$
$$= L_m(j) + xA - xA\,L_m(j)$$
$$= L_m(j) \oplus x\,[A_2 + \ldots + A_v]\,. \quad \blacklozenge$$

Together, Properties 2 and 3 show that certain "cancellations" in the update step (4) of the iterative scheme can be predicted in advance, and thus unnecessary computation can be avoided. The next property demonstrates that the approximations to $R_{sj}(G)$, derived from successive labels at node j, are monotone nondecreasing. The notation $R_m(j)$ indicates the value obtained by substituting numerical values p_r for x_r into the polynomial $L_m(j)$.

Property 4. If $k \le m$ then $R_k(j) \le R_m(j)$.

Proof. Since $k \le m$ we can express $L_k(j) = T_1 \oplus T_2 \oplus \ldots \oplus T_v$ and $L_m(j) = T_1 \oplus T_2 \oplus \ldots \oplus T_w$, where T_i is a monomial term representing some path P_i from s to j and $v \le w$. Then $R_m(j)$ represents the probability that at least one path of $\{P_1, P_2, \ldots, P_w\}$ is functioning and so is at least as large as the probability $R_k(j)$ that at least one path of $\{P_1, P_2, \ldots, P_v\} \subseteq \{P_1, P_2, \ldots, P_w\}$ is functioning. $\blacklozenge$

5. **Computational Results**

Several examples will be given in this section to illustrate the efficacy of a version of the iterative algorithm that makes use of Properties 2 and 3. The quality of the nondecreasing sequence of approximations to $R_{sj}(G)$ will also be examined, in particular as this relates to the discipline (FIFO, LIFO) used for managing the list L. The iterative

algorithm was coded in FORTRAN 77 and all computations were performed using the IBM 3081 computer at Clemson University.

Example 1. This network, having 9 nodes and 19 edges, is taken from [20] and is shown in Fig. 3. There are 35 s-t paths and 5287 noncancelling terms in $F_{st}(\mathbf{x})$. As discussed in Satyanarayana and Prabhakar [20], each noncancelling term corresponds to an "acyclic subgraph" of G. Despite its small size, this example represents one of the most complex directed networks whose exact reliability has been reported in the literature.

The reliabilities $R_{sj}(G)$ have been calculated using our iterative procedure and the FIFO/LIFO disciplines. For ease of presentation, the reliability polynomial $F_{st}(\mathbf{x})$ has been evaluated with all $p_k = p$ for the particular (s, t) pair indicated in Fig. 3; all edge failures are assumed to be independent. Fig. 4 shows $F(p) = F_{st}(p, ..., p)$ plotted versus p using the FIFO discipline. As expected, the various iterations produce an increasing sequence F1, ..., F9 of reliability curves that converge to the exact answer in 9 iterations. Each iteration produces a lower bound on $R_{st}(G)$ and thus provides a conservative estimate for the true network reliability. Namely, the exact (s, t)-reliability of the network is guaranteed to be at least as large as the value specified by the approximation. Notice that the curves for the fifth through ninth iterations overlap in the figure, thus providing excellent approximations to $R_{st}(G)$. Also indicated on Fig. 4 are the cumulative CPU times (in seconds) required to complete the work through the end of the specified iteration. Thus, a total of 0.638 seconds were needed to obtain $R_{st}(G)$, whereas only 0.061 seconds were needed to obtain an approximation that is virtually indistinguishable over the entire range $0 \le p \le 1$.

Fig. 5 shows analogous information relative to the LIFO discipline. In this case, twelve iterations were required before convergence was obtained. (Several of the curves overlap so only 10 approximations are apparent in the figure.) Although the exact answer was obtained in 0.454 seconds (less than the comparable time for FIFO), the

LIFO discipline did not give as useful a set of approximations compared to the FIFO approach.

Example 2. This network, with 13 nodes and 27 edges, is derived from an example given by Martelli [12]; see Fig. 6. It is considerably more complex than Example 1, having 70 s-t paths and 34,983 noncancelling terms. Plots of F(p) versus p are shown in Figs. 7 and 8 for the FIFO and LIFO disciplines, respectively. Again it is observed that the LIFO method obtains the exact answer faster than the FIFO method. However, the quality of approximations produced by FIFO is superior to those produced by LIFO. Indeed, a very close approximation to the exact reliability polynomial is obtained by FIFO in 1.18 seconds, one-eighth of the time required to find the exact answer using FIFO and one-sixth of that required using LIFO.

Finally, five random networks on 12 nodes and 30 edges were generated for test purposes. The characteristics of these networks, together with the number of iterations required for convergence, are shown in Table 2. In order to compare the quality of the approximations generated for these examples, we have tabulated the CPU time (in seconds) required to achieve a relative error of α% or less (at $p = 0.5$) in Table 3. The results for Examples 1 and 2 are also included.

In these random examples the FIFO and LIFO disciplines appear to be comparable in terms of the time required to obtain the exact answer. Again, however, the FIFO variant gives a fairly close approximation rather quickly and it completely dominates the LIFO variant in this respect.

Table 2

Network	#s-t Paths	#Noncancelling Terms	#Iterations FIFO	#Iterations LIFO
R1	14	1,263	9	7
R2	28	3,383	11	8
R3	41	7,583	8	10
R4	44	17,919	5	5
R5	34	42,687	10	8

Table 3

Example	Discipline	CPU (secs) for accuracy within α%			
		0%	1%	5%	10%
1	FIFO	.638	.248	.061	.033
	LIFO	.454	.454	.083	.083
2	FIFO	9.23	4.26	1.18	.277
	LIFO	6.92	6.92	6.92	.847
R1	FIFO	.049	.001	.001	.001
	LIFO	.054	.054	.004	.001
R2	FIFO	.380	.076	.006	.003
	LIFO	.362	.362	.362	.362
R3	FIFO	1.21	.172	.172	.004
	LIFO	1.11	.630	.162	.071
R4	FIFO	3.06	.195	.004	.004
	LIFO	2.85	2.85	.089	.005
R5	FIFO	5.07	.062	.002	.001
	LIFO	5.11	1.58	1.58	.002

6. Conclusions

This paper has explored an algebraic structure underlying certain network reliability problems. A promising iterative algorithm has been developed that allows both exact and approximate answers to be obtained. Rather than giving simply a single number, this algorithm produces a reliability polynomial that can then be easily evaluated at any particular input values $p_1, ..., p_r$. Also, in the process of determining $R_{st}(G)$ we also generate $R_{sj}(G)$ for all $j \in N$.

Empirical results have shown that the choice of data structure (FIFO, LIFO) can have a significant effect on the relative efficiency of the procedures as well as on the quality of the approximations. Whereas the LIFO approach frequently obtains the exact reliability polynomial faster than the FIFO approach, the latter produces better approximations -- ones that are quite close to the exact answer but are obtained in a fraction of the time. This desirable feature of the FIFO approach can be explained as follows, assuming that the p_k are comparable in value. Under a FIFO discipline, nodes are processed in order of increasing distance from s. Thus, the first time node j is labelled, it is done so relative to a path with the minimum number of edges. More generally, the FIFO approach ensures that the "more probable" (fewer edge) paths are incorporated as soon as possible. Subsequent (longer and less probable) paths contribute, but not as much, to the final label on node j. On the other hand, a LIFO discipline creates a depth-first rather than a breadth-first search of the network, and thus "early" approximations can be substantially improved by the incorporation of later (shorter) paths.

The approximate solutions generated by the iterative algorithm will always produce (conservative) lower bounds on the exact solution. If greater accuracy is required, such lower bounds can be used together with a simulation approach, such as Fishman's sampling procedure [7], that makes explicit use of lower bounds to obtain improved estimates. Alternatively, these lower bounds can be used in conjunction with existing

techniques that produce upper bounds on network reliability [22, 25] to obtain an interval that must enclose $R_{st}(G)$.

Finally, it should be emphasized that regardless of the list discipline used, some relatively challenging directed networks from the literature can be solved by our algorithm with a modest amount of computation. In particular, one example studied had 70 paths. We are not aware of any existing algorithm that has solved exactly a problem of this complexity. While the proposed approach appears to have potential, further experimentation will be necessary before any firm conclusions can be drawn concerning its general applicability. In order to solve larger, more realistic problems it may be possible to combine this approach with methods for decomposing the network into more manageable portions [23,26].

Acknowledgments

This paper benefited greatly from the careful scrutiny of the referees. This research was supported by the U. S. Air Force Office of Scientific Research (AFSC) under Grant AFOSR-84-0154. Reproduction in whole or in part is permitted for any purpose of the United States Government.

REFERENCES

[1] A. AGRAWAL AND R. E. BARLOW, A survey of network reliability and domination theory, Oper. Res., 32 (1984), pp. 478-492.

[2] A. AGRAWAL AND A. SATYANARAYANA, An O(|E|) time algorithm for computing the reliability of a class of directed networks, Oper. Res., 32 (1984), pp. 493-515.

[3] A. AGRAWAL AND A. SATYANARAYANA, Network reliability analysis using 2-connected digraph reductions, Networks, 15 (1985), pp. 239-256.

[4] M. O. BALL, Complexity of network reliability computations, Networks, 10 (1980), pp. 153-165.

[5] R. E. BARLOW AND F. PROSCHAN, Statistical Theory of Reliability and Life Testing, Holt, Rinehart and Winston, New York, 1975.

[6] B. CARRÉ, An algebra for network routing problems, J. Inst. Math. Appl., 7 (1971), pp. 273-294.

[7] G. S. FISHMAN, A monte carlo sampling plan for estimating reliability parameters and related functions, Technical Report UNC/ORSA/TR-85/7, University of North Carolina, Chapel Hill, 1985.

[8] M. GONDRAN AND M. MINOUX, Graphs and Algorithms, John Wiley & Sons, Chichester, 1984.

[9] C. L. HWANG, F. A. TILLMAN AND M. H. LEE, System-reliability evaluation techniques for complex/large systems - a review, IEEE Trans. Rel., R-30 (1981), pp. 416-422.

[10] Y. KIM, K. CASE AND P. GHARE, A method for computing complex system reliability, IEEE Trans. Rel., R-21 (1972), pp. 215-219.

[11] M. LOCKS, Recursive disjoint products: a review of three algorithms, IEEE Trans. Rel., R-31 (1982), pp. 33-35.

[12] A. MARTELLI, A gaussian elimination algorithm for the enumeration of cut sets in a graph, J. Assoc. Comp. Mach., 23 (1976), pp. 58-73.

[13] H. MINE, Reliability of physical system, IRE Trans. Circuit Theory, CT-6 (1959), pp. 138-151.

[14] T. POLITOF AND A. SATYANARAYANA, An $O(|V|^2)$ algorithm for a class of planar graphs to compute the probability that the graph is connected, Math. Oper. Res., to appear.

[15] T. POLITOF AND A. SATYANARAYANA, A linear time algorithm to compute the reliability of planar cube-free graphs, Technical Report, Department of Quantitative Methods, Concordia University, Quebec, 1985.

[16] J. S. PROVAN, The complexity of reliability computations in planar and acyclic graphs, Technical Report UNC/ORSA/TR-83/12, University of North Carolina, Chapel Hill, 1984.

[17] J. S. PROVAN AND M. O. BALL, The complexity of counting cuts and of computing the probability that a graph is connected, SIAM J. Comput., 12 (1983), pp. 777-788.

[18] A. ROSENTHAL, Computing the reliability of complex networks, SIAM J. Appl. Math., 32 (1977), pp. 384-393.

[19] A. SATYANARAYANA AND M. CHANG, Network reliability and the factoring theorem, Networks, 13 (1983), pp. 107-120.

[20] A. SATYANARAYANA AND A. PRABHAKAR, A new topological formula and rapid algorithm for reliability analysis of complex networks, IEEE Trans. Rel., R-27 (1978), pp. 82-100.

[21] A. SATYANARAYANA AND R. K. WOOD, A linear-time algorithm for computing K-terminal reliability in series-parallel networks, SIAM J. Comput., 14 (1985), pp. 818-832.

[22] J. G. SHANTHIKUMAR, Simple bounds for network reliability, Technical Report, School of Business Administration, University of California, Berkeley, 1984.

[23] D. R. SHIER, A decomposition algorithm for optimality problems in tree-structured networks, Discrete Math., 6 (1973), pp. 175-190.

[24] D. R. SHIER, Iterative algorithms for calculating network reliability, in Graph Theory with Applications to Algorithms and Computer Science, Y. Alavi et al. (eds.), John Wiley & Sons, 1985, pp. 741-752.

[25] A. SHOGAN, Sequential bounding of the reliability of a stochastic network, Oper. Res., 24 (1976), pp. 1027-1044.

[26] R. E. TARJAN, Fast algorithms for solving path problems, J. Assoc. Comp. Mach., 28 (1981), pp. 594-614.

[27] L. G. VALIANT, The complexity of enumeration and reliability problems, SIAM J. Comput., 8 (1979), pp. 410-421.

[28] R. K. WOOD, A factoring algorithm using polygon-to-chain reductions for computing K-terminal network reliability, Networks, 15 (1985), pp. 173-190.

END

FILMED

MARCH, 1988

DTIC